Further Reducing the Redundancy of a Notation over a Minimally Redundant Digit Set

Marc Daumas, David W. Matula

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Further reducing the redundancy of a notation over a minimally redundant digit set

Marc Daumas, CNRS

David W. Matula, SMU

No 3886

Février 2000

THÈME 2

ISSN 0249-6399 ISRN INRIA/RR--3886--FR+ENG

Further reducing the redundancy of a notation over a minimally redundant digit set

Marc Daumas, CNRS
David W. Matula, SMU

Thème 2 — Génie logiciel
et calcul symbolique
Projet Arénaire

Rapport de recherche n ° 3886 — Février 2000 — 20 pages

Abstract: Redundant notations are used implicitly or explicitly in many digital designs. They have been studied in details and a general framework is known to reduce the redundancy of a notation down to the minimally redundant digit set. We present here an operator to further reduce the redundancy of such a representation. It does not reduce the number of allowed digits since removing one digit to a minimally redundant digit set is a conversion to a non redundant digit set and this is an expensive operation. Our operator introduces some correlation between the digits to reduce the number of possible redundant notations for any represented number. This reduction is visible in small useful operators like the elimination of leading zeros. We also present a key application with a CMOS Booth recoded multiplier. Our multiplier is able to accept both a redundant or a non redundant input with very little modifications and almost no penalty in time or space compared to state-of-the-art non redundant multipliers.

Key-words: Redundant notation, Language, Multiplication, Computer arithmetic, Computer architecture, CMOS technology.

(Résumé : tsvp)

This work has been partially supported by the PICS 479 from the French National Center for Scientific Research (CNRS).

Authors may be reached via e-mail at Marc.Daumas@ENS-Lyon.Fr and Matula@SEAS.SMU.Edu. David Matula's permanent address is at the School of Engineering and Applied Sciences of the Southern Methodist University in Dallas, Texas.

This text is also available as a research report of the Laboratoire de l'Informatique du Parallélisme `http://www.ens-lyon.fr/LIP`.

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN (France)
Téléphone : 04 76 61 52 00 - International: +33 4 76 61 52 00
Télécopie : 04 76 61 52 52 - International: +33 4 76 61 52 52

Réduire encore la redondance d'une notation sur un ensemble de chiffres redondant minimal

Résumé : Les notations redondantes sont utilisées de façon implicite ou explicite dans de nombreux circuits numériques. Elles ont été étudiées en détail et des méthodes générales sont connues pour réduire la redondance jusqu'à atteindre un ensemble de chiffres redondant minimal. Nous présentons ici un opérateur pour réduire encore la redondance d'une telle notation. Il ne réduit pas le nombre de chiffres autorisés car supprimer un chiffre à une notation sur un ensemble de chiffres déjà redondant minimum est une conversion vers une notation non redondante et une opération longue. Notre opérateur introduit des correlations entre les chiffres pour réduire le nombre des notations redondantes possibles pour chaque nombre représenté. Cette réduction a un effet visible pour de petits opérateurs tels que l'élimination de zéros non significatifs en tête d'un mot. Nous présentons aussi une application importante avec un multiplieur CMOS basé sur le codage de Booth. Notre multiplieur est capable de traiter une entrée redondante ou non redondante avec très peu de modifications et sensiblement aucune pénalité en vitesse et en taille comparé à un multiplieur capable uniquement de traiter des nombres non redondants.

Mots-clé : Notation redondante, Langage, Multiplication, Arithmétique des ordinateurs, Architecture des ordinateurs, Technologie CMOS.

1 Introduction and Summary

Any positive integer has a unique radix β representation ($\beta \in \mathbb{N}$ and $\beta \geq 2$) with digits in the set $\{0, \cdots, \beta - 1\}$. When the set contains more than β elements, the notation is redundant and some integers have several representations. If the redundant representation of a number is computed by an automatic device such as a computer, we might be able to retrieve some correlations between the digits.

One such example is the radix 4 Booth recoding on the digit set $\{-2, \cdots, 2\}$ as presented in [1, 2] where a digit 2 can only be followed by a negative digit possibly preceded by a string of zeros. For example, $(201)_4 = 2 \cdot 4^2 + 0 \cdot 4^1 + 1 \cdot 4^0 = 33$ is not a valid Booth recoded representation. The valid one for 33 is $(1\overline{2}01)_4 = 1 \cdot 4^3 + (-2) \cdot 4^2 + 0 \cdot 4^1 + 1 \cdot 4^0$.

The conversion from the usual radix 2 representation to the radix 4 Booth recoded representation is obtained from the bit to digit conversion presented in Table 3, page 13. This special radix 4 notation on the redundant digit set $\{-2, \cdots, 2\}$ is not redundant. Contrary to general beliefs, converting a Booth recoded number to its conventional non redundant representation would not involve any carry ripple and could be performed in parallel (should any user be interested in doing it). This property is not visible by looking at the digit set alone.

In Section 2, we identify a quantity that describes part of the correlation of the digits. This quantity was first presented in [3] and it has been used in [4, 5, 6, 7, 8]. It is called the fraction range associated to a set of valid representations (language). It is the set of all the possible fractions of units in the last position lost when one truncates any valid representation for any possible position in the representation. This measure proves to be appropriate in retaining most of the information when carry recodings are applied. Such operations are specific digit set conversions [9] or rewriting rules [10, 11] that compute in parallel a ± 1 carry and an in-place residual digit for each position. Digit set conversion is a generalization of the addition of numbers in redundant and/or non redundant notations on parallel and serial systems [12, 13]. The fraction range traces the correlation of the digits in a complex arithmetic algorithm that uses only recodings as this is the case for most implemented arithmetic operators on a redundant notation.

We present in Section 3 three applications of redundancy reduction of radix two borrow save and carry save notations, one of them being a Booth multiplier with one redundant operand. Previous research has shown the feasibility of multiplier designs employing redundant binary operands. To avoid the general increase in hardware size entailed by redundant binary input, recent attention has been focused on limiting redundant input simply to the multiplier recoder input [14, 15, 16, 17, 18]. With minimum circuitry, we are ready to derive the low-power hot-one (only one signal is set at any time) signal controls { -2, -1, 0, 1, 2 } [19] or the common multiplier controls { negative; doubled factor; unchanged factor } [20, 21, 22]. But we prefer to compute the enhanced sign select controls { negative; positive; doubled factor; unchanged factor } [23]. For practical reasons, this seems more desirable than converting the number from digit set $\{0, \cdots, 3\}$ (non redundant input), $\{0, \cdots, 6\}$ (carry save input) or $\{-3, \cdots, 3\}$ (borrow save input) to digit set $\{-2, \cdots, 2\}$ before converting each of the digits obtained to control signals. Past recoders have added critical path delay for the

more frequent case where non redundant binary input is available. Our proposed circuit does not lengthen the time of one multiplication, compared to the state-of-the-art encoding if both inputs are non redundant.

2 Fraction range and carry recoding

2.1 Definitions

The radix β notation $(d_m \cdots d_0 \,.\, d_{-1} \cdots d_l)_\beta$ on the digit set S represents the rational number of equation (1). The set of all the possible representations from positions l through m with digits in S is the language noted S_l^m $(l \leq 0 \leq m)$. The valuation $||\cdot||_\beta$ maps S_l^m into the set $\mathbb{Q}$ of the rational numbers.

$$||d_m \cdots d_0 \,.\, d_{-1} \cdots d_l||_\beta = \sum_{i=l}^{m} d_i \beta^i \qquad \text{with} \qquad l \leq i \leq m \quad \Rightarrow \quad d_i \in S \tag{1}$$

In the following, we are only interested in contiguous digits sets containing zero, that is $S = \{S_{\min}, \cdots, S_{\max}\}$ with $S_{\min} \leq 0 \leq S_{\max}$ leading to a redundant notation as soon as $S_{\max} - S_{\min} \geq \beta$. The digit set is minimally redundant if $S_{\max} - S_{\min} = \beta$ and a carry ripple process with best time complexity $\Theta(\log(m-l))$ is necessary to reduce S to only β elements.

On an automatic device such as a computer circuit, a number is produced as a vector of digits $D = [d_m \cdots d_0 \; d_{-1} \cdots d_l]$. In the introduction, we have seen that there might be some correlations between the digits of the vector as long as they have not been given by the user but computed by an algorithm $A(X) = D$ to perform some arithmetic operations. Before knowing the actual value of the digits stored in the vector D, we can define some representations that are acceptable and prove that some others cannot occur. This defines the language $A^+ = \{A(X)\} \subset S_l^m$ which is the subset of all the possible output representations. By extension, we use the same notation for A^+ and A. We define the fraction range of A and the fraction range at position j by equations (2) and (3). We will see how to use the quantity and how this single set captures alone some very relevant part of the correlation between the digits.

$$Fr_j(A) = Fr_j(A^+) = \left\{ ||0 \,.\, d_{j-1} \cdots d_l||_\beta \;\;,\;\; [d_m \cdots d_l] \in A^+ \right\} \tag{2}$$

$$Fr(A) = Fr(A^+) = \bigcup_{l \leq j \leq m} Fr_j(A^+) \tag{3}$$

An input vector given by the user, where nothing is known about any of its digits, yields the largest *a priori* fraction range associated with this radix and this digit set. It is bounded

by equation (4)[1].

$$Fr(S_l^m) = \{||0\ .\ d_{j-1}\cdots d_l||_\beta \quad , \quad l \le j \le m \quad , \quad [d_m \cdots d_l] \in S_l^m\} \subset \frac{1}{\beta-1}\cdot(S_{\min}; S_{\max}) \cup \{0\} \tag{4}$$

For radix two representations, the fraction range of a borrow save vector where $S = \{-1; 0; 1\}$ and a carry save vector where $S = \{0; 1; 2\}$ are respectively $(-1, 1)$ and $[0, 2)$. Indeed this fact was recognized in Intel's description of the Pentium bug: truncating a number in carry save format produces an error within 2 ulps called the "region of uncertainty" [24, 25]. If the number is stored in borrow save format the error is within ± 1 ulp.

A positive carry recoding transforms a radix β notation $(d_m \cdots d_l)_\beta$ on the digit set $S = \{S_{\min}, \cdots, S_{\max}\}$ to the representation $(e_{m+1}\ e_m \cdots e_l)_\beta$. It uses a quantity $p \in S$ called the pivot to define the equations (5) and (6) for $l \le i \le m+1$ where $c_l = d_{m+1} = c_{m+2} = 0$.

$$c_{i+1} = \begin{cases} 1 & \text{if} \quad d_i \ge p \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

$$c_{i+1}\beta + e_i = d_i + c_i \tag{6}$$

We verify by induction on m that a recoding does not change the value stored as presented below.

$$\sum_{i=l}^{m} d_i\beta^i = \sum_{i=l}^{m+1} e_i\beta^i$$

The recoded digit set is $\{\min(S_{\min}; p-\beta), \cdots, \max(S_{\max}-\beta+1; p)\}$. The digit set is reduced if $S_{\min} + \beta \le p < S_{\max}$. The output digit set is minimally redundant as soon as $S_{\min} \ge p - \beta$ and $p + \beta > S_{\max}$, that is the case for example if the digit set is maximally redundant and $p = 0$. If the output digit set is minimally redundant, it is $\{p-\beta, \cdots, p\}$. If $p = S_{\max} = S_{\min} + \beta$ the output digit set is minimally redundant, identical to the input set.

A negative carry conversion is similar to the positive carry conversion except that c_{i+1} is defined from equation (7).

$$c_{i+1} = \begin{cases} -1 & \text{if} \quad d_i \le p \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

The recoded digit set is $\{\min(S_{\min}+\beta-1; p), \cdots, \max(S_{\max}; p+\beta)\}$. The digit set is reduced if $S_{\min} < p \le S_{\max} - \beta$. If the output digit set is minimally redundant, it is $\{p, \cdots, p+\beta\}$. It is unchanged, minimally redundant, if $p = S_{\min} = S_{\max} - \beta$. We will see that even when we do not reduce the digit set width, such conversions perform some legitimate work visible through its effect on the fraction range.

[1] The product of a set B by a scalar a is the set of any element of B scaled by a, $a \cdot B = \{a \cdot b\ ,\ b \in B\ \}$.

2.2 Fundamental property

If A is a language of S_l^m with fraction range $Fr(A)$ and P any positive carry recoding with output digit set $\{S_{\min}, \cdots, S_{\max}\}$ then[2]

$$Fr(P(A)) \subset \frac{1}{\beta}\left(Fr(A) + \{S_{\min}, \cdots, S_{\max} - 1\}\right) \tag{8}$$

Proof Let $D \in A$ with $D = [d_m \cdots d_l]$, we define the carry vector $C = [c_{m+1}\ c_m \cdots c_{l+1}]$ from equation (5) and the result vector $E = P(D) = [e_{m+1}\ e_m \cdots e_l]$ from equation (6). We obtain the following equation for any rational number $f \in Fr_j(P(A))$ by extracting a term from the radix polynomial of E from equation (1)

$$f = ||0\ .\ e_{j-1} \cdots e_l||_\beta = \frac{e_{j-1} - c_{j-1}}{\beta} + ||0\ .\ c_{j-1}\ e_{j-2} \cdots e_l||_\beta.$$

We recognize that $[c_{j-1}\ e_{j-2} \cdots e_l]$ is the recoded vector of $[d_{j-2} \cdots d_l]$ and therefore

$$||0\ .\ c_{j-1}\ e_{j-2} \cdots e_l||_\beta = ||0\ .\ 0\ d_{j-2} \cdots d_l||_\beta \in \frac{1}{\beta} Fr_{j+1}(A) \subset \frac{1}{\beta} Fr(A).$$

We know from the definition of the output digit set that $e_{j-1} - c_{j-1} = d_{j-1} - \beta c_j < S_{\max}$ to allow c_{j-1} to be incorporated without carry ripple. It follows that

$$Fr(P(A)) \subset \bigcup_{k=S_{\min}}^{S_{\max}-1} \frac{1}{\beta}\left(\{k\} + Fr(A)\right) = \frac{1}{\beta}\left(Fr(A) + \{S_{\min}, \cdots, S_{\max} - 1\}\right).$$

□

We prove in a very similar way that if N is a negative carry recoding then

$$Fr(N(A)) \subset \frac{1}{\beta}\left(Fr(A) + \{S_{\min} + 1, \cdots, S_{\max}\}\right). \tag{9}$$

Considering equations (8) and (9) it is now legitimate to perform a digit set recoding from one digit set to the same digit set since it reduces the fraction range of the final vector. We can recode a borrow save vector to reduce its fraction range from (a, b) to $\left(\frac{-1+a}{2}, \frac{b}{2}\right)$ with a positive recoding and to $\left(\frac{a}{2}, \frac{1+b}{2}\right)$ with a negative recoding. For a carry save number, we can reduce its fraction range from $[a, b)$ to $\left[\frac{a}{2}, \frac{1+b}{2}\right)$ with a positive recoding and to $\left[\frac{1+a}{2}, 1 + \frac{b}{2}\right)$ with a negative recoding.

We start with digits in the set $\{p - \beta, \cdots, p - 1 + r\}$ with $1 \leq r \leq \beta$ for a generalization of this last observation. Let P be a positive recodings with pivot p, then by induction

$$Fr(P^k) \subset \frac{1}{\beta - 1} \cdot \left(p - \beta, p - 1 + \frac{r}{\beta^k}\right) \cup \{0\}. \tag{10}$$

[2]The sum of two sets A and B is the set of the sum of any two elements from A and B, $A + B = \{\ a + b\ ,\ (a; b) \in A \times B\ \}$.

Comparing the fraction range of such a vector recoded k times to the fraction range of a non redundant vector $\frac{1}{\beta-1} \cdot (p-\beta; p-1)$, we see that the difference $r \cdot \beta^{-k}$ is decreasing geometrically with each new recoding. We can even apply another recoding to center the added quantity. Let N be the negative recoding with pivot $p-\beta$, the fraction range $NP^k(X)$ is bounded by equation (11). The centering is best if $\beta = 2$ since $\frac{\beta-1}{\beta} = \frac{1}{\beta} = \frac{1}{2}$.

$$Fr(NP^k) \subset \frac{1}{\beta-1} \cdot \left(p-\beta+\frac{\beta-1}{\beta}; p-1+\frac{\beta-1}{\beta}+\frac{r}{\beta^{k+1}}\right) \cup \{0\}. \tag{11}$$

Proof The proof of equation (10) is easily done by induction on k. We prove it for $k=1$. From equation (4), we know that $Fr(S_l^m) \subset \frac{1}{\beta-1} \cdot (p-\beta, p-1+r) \cup \{0\}$. The output digit set is minimally redundant since $p-1+r < p+\beta$ and equation (8) gives us the inclusion:

$$\begin{array}{rcl} Fr(P) & \subset & \frac{1}{\beta}\left(Fr(S_l^m) + \{p-\beta, \cdots, p-1\}\right) \\ & & \frac{1}{\beta}\left(\frac{1}{\beta-1}(p-\beta)+p-\beta; \frac{1}{\beta-1}(p-1+r)+p-1\right) \cup \{0\} \\ & & \frac{1}{\beta-1}\left(p-\beta; p-1+\frac{r}{\beta}\right) \cup \{0\} \end{array}$$

Equation (11) is proved in a similar way:

$$\begin{array}{rcl} Fr(NP^k) & \subset & \frac{1}{\beta}\left(Fr(P^k) + \{p-\beta+1, \cdots, p\}\right) \\ & & \frac{1}{\beta}\left(\frac{1}{\beta-1}(p-\beta)+p-\beta+1; \frac{1}{\beta-1}(p-1+\frac{r}{\beta^k})+p\right) \cup \{0\} \\ & & \frac{1}{\beta-1}\left(p-\beta+\frac{\beta-1}{\beta}; p-1+\frac{\beta-1}{\beta}+\frac{r}{\beta^{k+1}}\right) \cup \{0\} \end{array}$$

□

We obtain an identical property if k negative recodings are applied to a vector of digits in the set $\{p+1-r, \cdots, p+\beta\}$ with $1 \leq r \leq \beta$. In this case N^k has its fraction range bounded by equation (12) and PN^k with one last positive recoding of pivot $p+\beta$ has its fraction range bounded by equation (13)

$$Fr(N^k) \subset \frac{1}{\beta-1} \cdot \left(p+1-\frac{r}{\beta^k}, p+\beta\right) \cup \{0\} \tag{12}$$

$$Fr(PN^k) \subset \frac{1}{\beta-1} \cdot \left(p+1-\frac{\beta-1}{\beta}-\frac{r}{\beta^{k+1}}; p+\beta-\frac{\beta-1}{\beta}\right) \cup \{0\} \tag{13}$$

2.3 First application: Combining several digits in a window

We will see Section 3 how the fraction range of a binary encoding can be used to deduce important properties on the digits stored in a vector. We will just present here a first practical application of the fraction range.

Let $D = [d_m \cdots d_l]$ be a vector of radix β digits. The window of k digits starting at position $j \cdot k$ can be valued alone as an integer d'_j given by equation (14). If we apply this

Table 1: Reducing the redundancy to obtain a minimally redundant digit set radix 10^2.

Rec.	Fraction range (Radix 10)	Digits radix 100	
		Max	Digit set
Non	$\frac{1}{9}\cdot(-1,8+1)=\left(-\frac{1}{9},1\right)$	99	$\{-11,\cdots,99\}$
P	$\frac{1}{9}\cdot\left(-1;8+\frac{1}{10}\right)=\left(-\frac{1}{9};\frac{9}{10}\right)$	$90+\frac{1}{9}$	$\{-11,\cdots,90\}$
P^2	$\frac{1}{9}\cdot\left(-1;8+\frac{1}{100}\right)=\left(-\frac{1}{9};\frac{89}{100}\right)$	$89+\frac{1}{9}$	$\{-11,\cdots,89\}$[3]
P^3	$\frac{1}{9}\cdot\left(-1;8+\frac{1}{1000}\right)=\left(-\frac{1}{9};\frac{889}{1000}\right)$	$88.9+\frac{1}{9}$	$\{-11,\cdots,89\}^3$

transformation for any $\lfloor m/k\rfloor \leq j \leq \lfloor l/k\rfloor$ we obtain a new vector of digits d'_j representing the same number radix β^k as shown in equation (15). We commonly use the octal and the hexadecimal notations because this conversion is very easy to perform back and forth from radix 2 with $k=3$ or $k=4$.

$$d'_j = ||d_{(j+1)\cdot k-1}\cdots d_{j\cdot k}||_\beta = \sum_{i=0}^{k-1} d_{j\cdot k+i}\beta^i \quad \text{with} \quad d'_j \in \frac{\beta^k-1}{\beta-1}\cdot[S_{\min};S_{\max}] \tag{14}$$

$$||d_m\cdots d_l||_\beta = \left|\left|d'_{\lfloor\frac{m}{k}\rfloor}\cdots d'_{\lfloor\frac{l}{k}\rfloor}\right|\right|_{\beta^k} \tag{15}$$

We can also write d'_j as presented in equation (16) leading to another interval for d'_j.

$$d'_j = ||0\,.\,d_{(j+1)\cdot k-1}\cdots d_l||_\beta\cdot\beta^k - ||0\,.\,d_{j\cdot k}\cdots d_l||_\beta \quad \text{with} \quad d'_j \in \left(\beta^k\cdot Fr(D) - Fr(D)\right) \tag{16}$$

If the fraction range satisfies $Fr(D)\subset(a,b)$, equation (16) yields that a combined digit d'_j is bounded by $a\cdot\beta^k - b < d'_j < b\cdot\beta^k - a$. We present Tables 1 and 2 two examples radix 10, with the digit set $\{-1,\cdots,9\}$ and up to 3 positive carry recodings with pivot $p=9$. The lower bound on the combined digit set is given by equation (14) and does not change. The maximum value for one digit is computed from the fraction range in the table. The first recoding removed approximately a portion $\frac{1}{\beta}$ of the possible digits and the second one a portion $\frac{1}{\beta^2}$.

Getting rid of almost $\frac{1}{\beta}$ of the digits allows us to reach easily the minimally redundant digit set. It may not seem to be important working radix 10 but this is very relevant radix 2 as we will see in the next section.

[3]The digit set is minimally redundant. There will be no further improvement.

Table 2: Reducing the redundancy to obtain a minimally redundant digit set radix 10^3.

Rec.	Fraction range (Radix 10)	Digits radix 1000	
		Max	Digit set
Non	$\frac{1}{9} \cdot (-1, 8+1) = \left(-\frac{1}{9}, 1\right)$	999	$\{-111, \cdots, 999\}$
P	$\frac{1}{9} \cdot \left(-1; 8+\frac{1}{10}\right) = \left(-\frac{1}{9}; \frac{9}{10}\right)$	$900+\frac{1}{9}$	$\{-111, \cdots, 900\}$
P^2	$\frac{1}{9} \cdot \left(-1; 8+\frac{1}{100}\right) = \left(-\frac{1}{9}; \frac{89}{100}\right)$	$890+\frac{1}{9}$	$\{-111, \cdots, 890\}$
P^3	$\frac{1}{9} \cdot \left(-1; 8+\frac{1}{1000}\right) = \left(-\frac{1}{9}; \frac{889}{1000}\right)$	$889+\frac{1}{9}$	$\{-111, \cdots, 889\}^3$

3 Binary applications

As we have seen in equations (10), (11), (12) and (13), a short sequence of recodings can be employed to reduce the fraction range and approach the width of 1 ulp. The practical value of binary carry recodings is that just a few recodings provide partial compression sufficient to obtain almost all the benefits of full conversion to a non redundant notation while avoiding the high cost of a carry-ripple addition. The following observations are straightforward from the definition and support partial compression applications in rounding, leading insignificant digit deletion, and multiplier recoding.

3.1 Implementation and bit specialization

In computers, the digit vector is not stored but encoded into bits. In the carry save format, each digit d_i is stored with two bits p_i and q_i and the digit value is defined as $d_i = p_i + q_i$. We specialize the bits of the result register $e_i = p'_i + q'_i$ such that p'_{i+1} stores the carry bit c_{i+1} generated by equation (5) and q'_i stores the residual quantity $e_i - c_i = d_i - c_{i+1}\beta$ of equation (6). This lead us to the truth table and the equations for the positive carry recoding of a carry save number Figure 1-a. If the numbers are stored using two's complement, the sign digit is moved to position $m+1$ and $q_{m+1} = q_m$. The equations of Figure 1-a are the ones of an half adder cell (HA) as we will see soon.

The negative carry recoding of a carry save number cannot be defined in such an elementary manner but we may define more recodings if the target encoding is borrow save instead of carry save. A borrow save digit d_i (resp. e_i) is stored with two bits p_i and n_i (resp. p'_i and n'_i) and the digit is defined as $d_i = p_i - n_i$ (resp. $e_i = p'_i - n'_i$). We specialize a positive carry or a negative carry recoding since we can store the carry in p'_{i+1} (positive carry P-recoding of Figure 1-b) or n'_{i+1} (negative carry N-recoding of Figure 1-c).

CS in	p_i	0	0	1	1
	q_i	0	1	0	1
Digit d_i		0	1	1	2
CS out	p'_{i+1}	0	0	0	1
	q'_i	0	1	1	0

Carry $p'_{i+1} = p_i \cdot q_i$
Residual $q'_i = p_i \oplus q_i$

(a) Positive carry recoding of a carry save number (HA)

BS in	p_i	0	0	1	1
	n_i	0	1	0	1
Digit d_i		0	-1	1	0
BS out	p'_{i+1}	0	0	1	0
	n'_i	0	1	1	0

Carry $p'_{i+1} = p_i \cdot \overline{n}_i$
Residual $n'_i = p_i \oplus n_i$

(b) Positive carry recoding of a borrow save number (P)

BS in	p_i	0	0	1	1
	n_i	0	1	0	1
Digit d_i		0	-1	1	0
BS out	p'_i	0	1	1	0
	n'_{i+1}	0	1	0	0

Carry $n'_{i+1} = \overline{p}_i \cdot n_i$
Residual $p'_i = p_i \oplus n_i$

(c) Negative carry recoding of a borrow save number (N)

CS in	p_i	0	0	1	1
	q_i	0	1	0	1
Digit d_i		0	1	1	2
BS out	p'_{i+1}	0	1	1	1
	n'_i	0	1	1	0

Carry $p'_{i+1} = p_i + q_i$
Residual $n'_i = p_i \oplus q_i$

(d) Negative carry recoding from carry save number to borrow save representation (Q)

Figure 1: Truth table and equations of binary recodings with a specialized carry bit

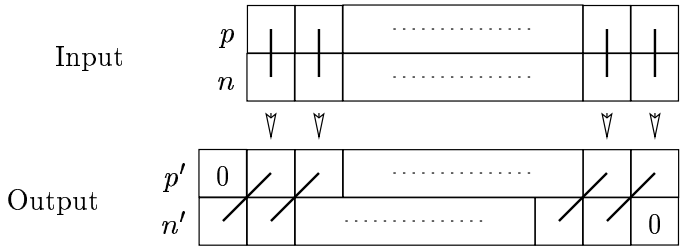


Figure 2: N-carry recoding of a borrow save number

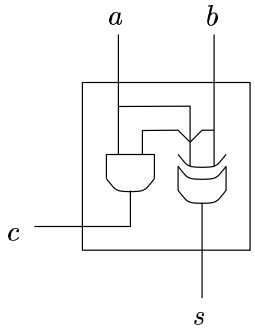
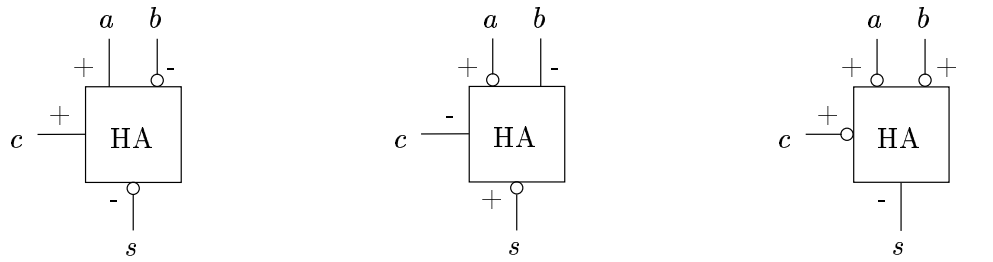


Figure 3: Half adder cell (HA) $a + b = 2c + s$

The positive carry recoding of a carry save number to borrow save notation is not possible but a negative carry recoding is possible. It is presented Figure 1-d as Q-recoding. Figure 2 illustrates how each output diagonal of the $2 \times (m - l + 2)$ bit array is determined by an input column of the $2 \times (m - l + 1)$ bit array for the N-carry recoding.

The half-adder cell is one of the basic cells of the computer arithmetic libraries. Functionally, it is implemented with an `exclusive or` gate and a `logical and` (see Figure 3). In choosing the place to add some new `logical inverters`, we define the bit level cells corresponding to the P-, N- and Q- recodings (see Fig 4). These inverters may or may not yield an actual penalty compare to a standard half adder cell. For example, the `exclusive or` gate may be implement using pass transistors as it is the case in state-of-the-art circuit design [23]. If so, an inverted `exclusive or` gate is obtained by switching some of the input wires of a straight `exclusive or` gate.

We will see in the remaining of the text that being an high level quantity, the fraction range can be easily tracked through an algorithm and is therefore very useful when more than one recoding is performed. Our redundant Booth recoder (section 3.3) uses some additional information on the last recoding.



(a) P-cell $a - b = 2c - s$

(b) N-cell $a - b = -2c + s$

(c) Q-cell $a + b = 2c - s$

Figure 4: Recoder cells

3.2 Rounding and leading digit deletion

If we truncate the low order part of a word, we may cause an error up to ± 1 ulp or 2 ulps depending on the notation. On the other hand, if we first apply a PN^{k-1} transformation ($k \geq 1$) to the borrow save register or a $PN^{k-2}Q$ transformation ($k \geq 2$) to a carry save register, we obtain a number stored in borrow save format that has a very limited fraction range included in $[-\frac{1}{2}, \frac{1}{2} + 2^{-k})$. As a consequence, truncating it at any position will result in an error less than $\frac{1}{2} + 2^{-k}$ ulp.

Three lines of half adder provides a PN^2 rounding with fraction range $(-\frac{5}{8}, \frac{1}{2})$ sufficient to reduce truncation error below $\frac{5}{8}$ ulps. This can be quite useful in microcoded redundant binary designs for division, square root, and transcendentals. The next observation has great applicability in extracting differences from a function table for performing interpolation [26, 27]. The $2 \times (m-l+1)$ bit array may be formed as the difference $a-b$ of two $(m-l+1)$-bit unsigned integers and stored in borrow save format with no operation. In this case, two recodings allow to discard the leading digits.

Partial compression also realizes virtually all the benefits of leading digit deletion. Let X be a positive number represented in borrow save such that $X \leq 2^k - 1$. We know that if $N(X) = [d'_{m+1}\ d'_m \cdots d'_l]$, then $d'_k = 0$ for all $i \geq k+1$. Thus $N(X)$ may be truncated to a $2 \times (k+1)$ bit array. If we can only bound the magnitude of X (*ie.* $|X| \leq 2^k - 1$), we have to compute $PN(X)$ and we prove that $d''_i = 0$ for all $i \geq k+2$

3.3 Booth Recoder

A radix 4 Booth multiplier computes in three steps the representation of $D = A \times B$ where the two numbers A and B are represented radix 2 by $(a_m \cdots a_l)_2$ and $(b_{m'} \cdots b_{l'})_2$. This organization is visible Figure 6 and in [20, 21, 23]. The first step converts B to a radix-4 minimally redundant recoding on the digit set $\{-2, \cdots, 2\}$ ("Booth encoding"). This conversion is performed for $\lfloor l'/2 \rfloor \leq j \leq \lceil m'/2 \rceil$ by looking at bits b_{2j+1}, b_{2j} and b_{2j-1} to

Table 3: Radix 4 Booth recoding

Multiplier representation			Booth digit	Sign select from [23]			
b_{2j+1}	b_{2j}	b_{2j-1}	$b'_j = -2b_{2j+1} + b_{2j} + b_{2j-1}$	X_j	TX_j	PL_j	M_j
0	0	0	0	0	1	0	0
0	0	1	1	1	0	1	0
0	1	0	1	1	0	1	0
0	1	1	2	0	1	1	0
1	0	0	-2	0	1	0	1
1	0	1	-1	1	0	0	1
1	1	0	-1	1	0	0	1
1	1	1	0	0	1	0	0

compute digit b'_j as presented in Table 3. One can check by induction that this operation does not change the represented value as written in the first part of equation (17). The second step ("Tree reduction") accumulates the partial products $b'_j \cdot A \cdot 4^j$ in a redundant format to compute the product as suggested by the second part of equation (17). The third step converts the redundant result to the usual non redundant binary representation and possibly round it according to the active rounding mode ("2-1 compression").

$$\sum_{i=l}^{m} b_i 2^i = \sum_{j=\lfloor l/2 \rfloor}^{\lceil m/2 \rceil} b'_j 4^j \quad \text{and} \quad A \times B = \sum_{i=l}^{m} b_i \cdot A \cdot 2^i = \sum_{j=\lfloor l/2 \rfloor}^{\lceil m/2 \rceil} b'_j \cdot A \cdot 4^j \tag{17}$$

Compared to the usual multiplication, Booth recoding divides by two the number of partial products generated and accumulated but these products are more difficult to generate since they cannot be obtained by a `logical and` gate as this is the case when we multiply only by $b_i \in \{0; 1\}$. A naive implementation can ruin all the advantages of Booth recoding. State of the art implementations usually present two cells: the encoder that is responsible of generating a set of control signals from the input bits b_{2j+1}, b_{2j} and b_{2j-1} and the multiplexer that computes a representation of $b'_j \cdot A$ based on $(a_m \cdots a_l)_2$ and the control signals generated by the encoder. As noted in [23], an IEEE-754 standard [28] double precision multiplier uses 27 encoders and 1527 multiplexers — up to 90% of the area of the circuit in some earlier publication [21].

We present in Table 3 the set of control signals compatible with the multiplexer used in [23]. The number obtained is the one's complement representation of $A \times b'_j$. Little extra circuitry is added to take care of 2's complement logic [2].

The next result supports the factoring of multiplier recoding into a two step process. Partial compression is first applied to recode a redundant format so in a second step it may be passed through a Booth recoder with very few penalty. The leading negative weight bit of any $2 \times k$ bit window of a borrow save number $PN^k(X)$ indicates the sign of the digit

value of that window whenever that digit is non-zero. The same result applies for a window on a $PN^{k-1}Q$ recoded number.

Proof Let $D = N^k(X)$ with $D = [d_m \cdots d_l]$, we define the carry vector $C = [c_{m+1}\ c_m \cdots c_{l+1}]$ from equation (5) and the result vector $E = P(D) = [e_{m+1}\ e_m \cdots e_l]$ from equation (6) with $e_i = p_i - n_i$. We obtain the following equation for any combined digit e'_i radix 2^k

$$e'_i = ||0\ .\ e_{(i+1)\cdot k-1} \cdots e_l||_2 \cdot 2^k - ||0\ .\ e_{i\cdot k} \cdots e_l||_2.$$

We extract a term from the radix polynomial of E from equation (1)

$$e'_i = (e_{(i+1)\cdot k-1} - c_{(i+1)\cdot k-1}) \cdot 2^{k-1} + ||0\ .\ c_{(i+1)\cdot k-1}\ e_{(i+1)\cdot k-2} \cdots e_l||_2 \cdot 2^k - ||0\ .\ e_{k\cdot n} \cdots e_l||_2.$$

We recognize that $[c_{(i+1)\cdot k-1}\ e_{(i+1)\cdot k-2} \cdots e_l]$ is the recoded vector of $[d_{(i+1)\cdot k-2} \cdots d_l]$ and $e_{(i+1)\cdot k-1} - c_{(i+1)\cdot k-1} = -n_{(i+1)\cdot k-1}$ and therefore

$$e'_i \in -n_{(i+1)\cdot k-1} \cdot 2^{k-1} + 2^{k-1} \cdot \left(-2^{-k}; 1\right) - \frac{1}{2} \cdot \left(-1 - 2^{-k}; 1\right)$$

For any k, if $n_{(i+1)\cdot k-1} = 1$ then $e'_i \leq -2^{k-1} + 2^{k-1} + \frac{1+2^{-k}}{2}$ and $e'_i \leq 0$ since $e'_i \in \mathbb{Z}$.

□

We can now list all the possible output of two digits e_{2j+1} and e_{2j} of a PN^2 or a PNQ recoder. As presented Table 4 some outputs are not valid because they violate the fraction range or the last result just presented or because they cannot be obtained from the last P recoding. We have also listed Table 4 all the valid cases if we store b_{2j+1} in n_{2j+1}, b_{2j} in p_{2j+1} and n_{2j} and finally b_{2j-1} in p_{2j}. The value of the stored number is unchanged since we virtually compute $2B - B$ as presented in details by the authors in [3].

We design the new encoder presented Figure 5-b to produce the correct result for each acceptable input of d_{2j+1} and d_{2j}. It shows how a borrow save register can be used as the input to the Booth encoding logic to store either a PNQ precoded carry save result or a non redundant number which has been converted to borrow save using the centered conversion. The modified Booth multiplier of Figure 6 accepts a redundant number as one of its operands, by the use of a precoder which does not generate any additional delay for a non-redundant operand.

4 Conclusion

We have presented a general formalism for the study of partial compression and roundings. This approach proves fruitful in reducing the redundancy of a borrow save or a carry save bit array to allow radix 2^k Booth recoding with minimal circuitry. Reducing redundancy is useful in any applications that do not allow any full range redundant number as input but do not require non redundant inputs.

Table 4: Radix 4 PN^2 or PNQ recoded borrow save number

Input bits				Radix 4	Acceptable digit?	
p_{2j+1}	n_{2j+1}	p_{2j}	n_{2j}	digit	Recoded	Non redundant
0	0	0	0	0	Yes	Yes
0	0	0	1	-1	No	No
0	0	1	0	1	Yes	Yes
0	0	1	1	0	Yes	No
0	1	0	0	-2	Yes	Yes
0	1	0	1	-3	No	No
0	1	1	0	-1	Yes	Yes
0	1	1	1	-2	Yes	No
1	0	0	0	2	No	No
1	0	0	1	1	Yes	Yes
1	0	1	0	3	No	No
1	0	1	1	2	Yes	Yes
1	1	0	0	0	No	No
1	1	0	1	-1	Yes	Yes
1	1	1	0	1	No	No
1	1	1	1	0	Yes	Yes

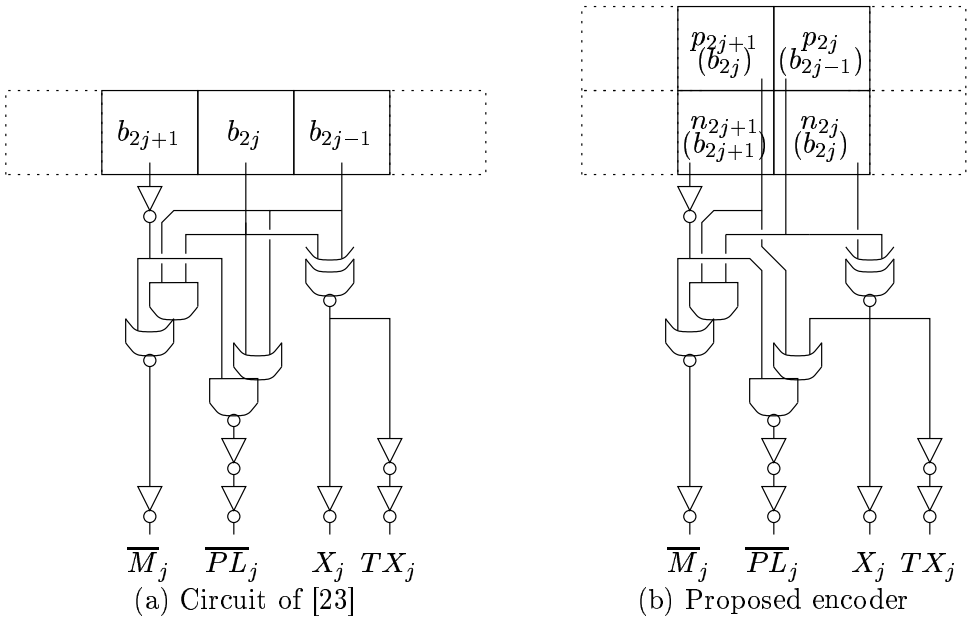


Figure 5: Usual non redundant and redundant aware enhanced sign select Booth encoders

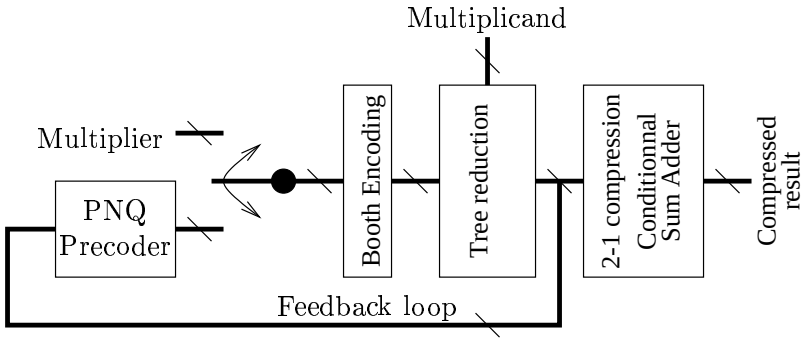
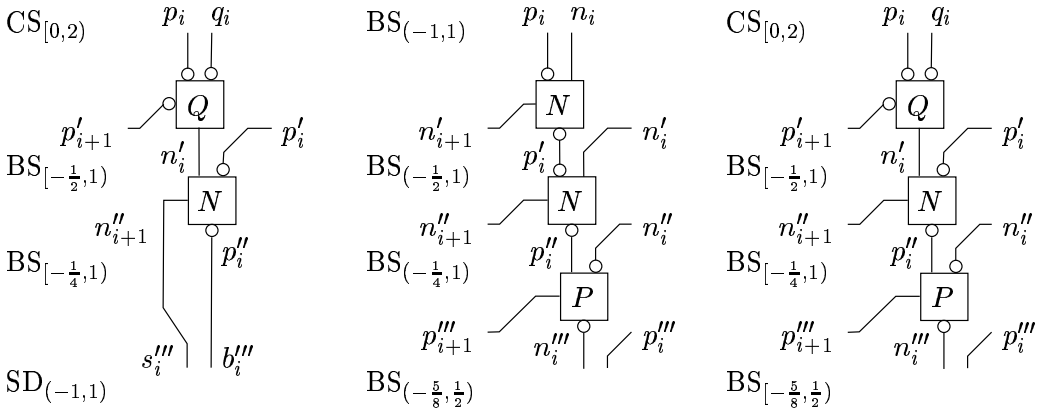


Figure 6: General purpose Booth multiplier with fast feedback capacities through a precoder



(a) Translated NQ-recoding (b) PN^2-recoding (c) PNQ-recoding

Figure 7: Useful transformations

We are able to build three useful recoder segments with our analysis (see Fig.7). To obtain the corresponding recoder, the desired segment is replicated k times, where k is the desired input length. Modified versions of the segment are used at the ends, to avoid using logic simply to generate constants. These precoders allow transformation of carry save to sign digit (see fig. 7-a and [3] for a more detailed discussion on signed digit notation) and of borrow save or carry save to radix 4 Booth recoding (Fig. 7-b and 7-c). The format and the fraction range are indicated after each recoding.

With the technology used in the design presented in [23], that is 0.25 μm CMOS technology with 2.5 V power supply. The authors obtain a 54×54 multiplier with a clock cycle of 4.1 ns. From spice simulations presented [23] we can predict that, each row of modified half-adders will delivers its output with a delay between 100 ps and 200 ps (depending on electrical properties). As a result, the circuit presented Figure 6 can perform a non redundant multiplication in 4.1 ns or prepare a redundant result to be reused by another multiplication in 3.4 ns. For example computing a product of 10 numbers will take 41 ns with the original multiplier and 35 ns with our modified one. Higher savings could be assessed by an electrical simulation of the PNQ recoding.

Other applications arise in the forwarding and feedback of number internal to a floating point unit to reduce redundancy as presented in [4]. Precoders are used to forward some parts of the redundant result after redundancy is sufficiently reduced. This allows an IEEE standard behavior of the rounding unit albeit the numbers are not compressed to a non redundant format. The construction of a standard adder using this technique is depicted in [5].

Other applications will appear in converting the two bounds of an interval to the step of a linear interpolation as it is performed for fast reciprocal and square root approximation [26]. This will allow to forward directly the couple of recoded bounds in borrow save format to a modified Booth multiplier.

5 Acknowledgment

We wish to thank the scientific community of the IEEE Symposium on Computer Arithmetic for their comments on this subject and specially Peter Kornerup for his notations as presented in [9] and his kind and knowledgeable help over the time of this work.

References

[1] A. D. Booth, "A signed binary multiplication technique," *Quarterly Journal of Mechanics and Applied Mathematics*, vol. 4, no. 2, pp. 236–240, 1951.

[2] I. Koren, *Computer Arithmetic Algorithms*. Prentice Hall, 1993.

[3] M. Daumas and D. W. Matula, "Recoders for partial compression and rounding," Research report 97-01, Laboratoire de l'Informatique du Parallélisme, Lyon, France, 1997.

[4] D. W. Matula and A. Munk Nielsen, "Pipelined packet-forwarding floating point: I. foundations and a rounder," in *Proceedings of the 13th Symposium on Computer Arithmetic* (T. Lang, J.-M. Muller, and N. Takagi, eds.), (Monterey, California), pp. 140–147, IEEE Computer Society Press, 1997.

[5] A. Munk Nielsen, D. W. Matula, C. N. Lyu, and G. Even, "Pipelined packet-forwarding floating point: I. an adder," in *Proceedings of the 13th Symposium on Computer Arithmetic* (T. Lang, J.-M. Muller, and N. Takagi, eds.), (Monterey, California), pp. 148–155, IEEE Computer Society Press, 1997.

[6] A. Munk Nielsen, D. W. Matula, C. N. Lyu, and G. Even, "An IEEE compliant floating point adder that conforms with the pipelined packet forwarding paradigm," *IEEE Transactions on Computers*, vol. 49, no. 4, pp. 33–47, 2000.

[7] P. M. Seidel and G. Even, "How many logic levels does floating-point addition require?," in *1998 International Conference on Computer Design*, (Austin, Texas), pp. 142–149, 1998.

[8] P. M. Seidel, "High speed redundant reciprocal apprximation," in *3rd Real Numbers and Computers Conference*, (Paris, France), pp. 219–229, 1998.

[9] P. Kornerup, "Digit-set conversion: generalizations and applications," *IEEE Transactions on Computers*, vol. 43, no. 5, pp. 622–629, 1994.

[10] T. M. Carter and J. E. Robertson, "The set theory of arithmetic decomposition," *IEEE Transactions on Computers*, vol. 39, no. 8, pp. 993–1005, 1990.

[11] M. Ercegovac and T. Lang, "On recoding in arithmetic algorithms," *Journal of VLSI Signal Processing*, vol. 14, pp. 283–294, 1996.

[12] A. Avižienis, "Signed digit number representations for fast parallel arithmetic," *IRE Transactions on Electronic Computers*, vol. 10, pp. 389–400, 1961.

[13] D. W. Matula, *Applied Computation Theory: Analysis, Design, Modeling*, ch. Radix Arithmetic: Digital Algorithms for Computer Architecture, pp. 374–448. Prentice Hall, 1976.

[14] N. Takagi and S. Yajima, "A fast iterative multiplication method by recoding intermediate product," in *Proceedings of the 36th National Convention of Information Science*, (Kyoto, Japan), 1987.

[15] N. Takagi, "Arithmetic unit based on a high speed multiplier with a redundant binary addition tree," in *Advanced Signal Processing Algorithms , Architectures and Implementation II*, vol. 1566 of *Proceedings of SPIE*, pp. 244–251, 1991.

[16] W. S. Briggs and D. W. Matula, "Rectangular array signed digit multiplier," US Patent 5 184 318, US Patent Office, 1993.

[17] B. W. Y. Wei, H. Du, and H. Chen, "A complex number multiplier using radix 4 digits," in *Proceedings of the 12th Symposium on Computer Arithmetic* (S. Knowles and W. H. McAllister, eds.), (Bath, England), pp. 84–90, IEEE Computer Society Press, 1995.

[18] C. N. Lyu and D. W. Matula, "Redundant binary Booth recoding," in *Proceedings of the 12th Symposium on Computer Arithmetic* (S. Knowles and W. H. McAllister, eds.), (Bath, England), pp. 50–57, IEEE Computer Society Press, 1995.

[19] R. M. Jessani and M. Putrino, "Comparison of single and dual pass multiply add fused floating point units," *IEEE Transactions on Computers*, vol. 47, no. 9, pp. 927–937, 1998.

[20] J. Mori *et al.*, "A 10 ns 54 × 54b parallel structured full array multiplier with 0.5 μm cmos technology," *IEEE Journal of Solid-State Circuits*, vol. 26, no. 4, pp. 600–605, 1991.

[21] G. Goto, T. Sato, M. Nakajima, and T. Sukemura, "A 54 × 54b regularly structured tree multiplier," *IEEE Journal of Solid-State Circuits*, vol. 27, no. 9, pp. 1229–1236, 1992.

[22] H. Makino *et al.*, "An 8.8ns 54 × 54-bit multiplier with high speed redundant binary architecture," *IEEE Journal on Solid State Circuits*, vol. 31, no. 6, 1996.

[23] G. Goto *et al.*, "A 4.1ns compact 54 × 54b multiplier utilizing sign select Booth encoders," *IEEE Journal of Solid-State Circuits*, vol. 32, no. 11, pp. 1676–1682, 1997.

[24] J.-M. Muller, "Algorithmes de division pour microprocesseurs : illustration à l'aide du "bug" du Pentium," *Technique et Science Informatiques*, vol. 14, no. 8, 1995.

[25] H. P. Sharangpani and M. L. Barton, "Statistical analysis of floating point flaw in pentium processors," White Paper 11, Intel Corporation, 1994.

[26] D. Das Sarma and D. W. Matula, "Hardware reciprocal table compression/decompression techniques," in *Scientific Computing and Validated Numerics* (G. Alefeld, A. Frommer, and B. Lang, eds.), (Wuppertal, Germany), pp. 11–17, Akademic Verlag, 1995.

[27] M. J. Schulte and J. E. Stine, "Approximating elementary functions with symmetric bipartite tables," *IEEE Transactions on Computers*, vol. 48, no. 8, pp. 842–847, 1999.

[28] D. Stevenson *et al.*, "An american national standard: IEEE standard for binary floating point arithmetic," *ACM SIGPLAN Notices*, vol. 22, no. 2, pp. 9–25, 1987.

[29] H. M. Darley *et al.*, "Floating point/integer processor with divide and square root functions," US Patent 4 878 190, US Patent Office, 1989.

[30] P. Chai *et al.*, "A 120 MFLOPS CMOS floating point processor," in *Proceedings of the 1991 Custom Integrated Circuits Conference*, (San Diego, California), pp. 15.1.1–15.1.4, IEEE Computer Society Press, 1991.

[31] H. Kabuo *et al.*, "Accurate rounding scheme for the Newton Raphson method using redundant binary representation," *IEEE Transactions on Computers*, vol. 43, no. 1, pp. 43–51, 1994.

[32] S. M. Quek, L. Hu, J. P. Prabhu, and F. A. Ware, "Apparatus for determining Booth recoder input control signals," US Patent 5 280 439, US Patent Office, 1994.

Unit´e de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unit´e de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unit´e de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unit´e de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unit´e de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
http://www.inria.fr
ISSN 0249-6399